Application of Fuzzy Control for Building Energy Management.

Pierre Yves Glorennec
Département d'Informatique
INSA de Rennes
35043 Rennes Cedex
glorenne@irisa.fr
France

Abstract

In this paper, we want to show an application of fuzzy control to building thermal regulation. Thanks to a new learning method, inspired from connexionnist techniques, the controller learns to identify the rules it must use, either without any previous knowledge, or with approximative rules. This method is tested in simulation on a thermal regulation problem and is compared to a discrete time PI controller.

Keywords

fuzzy logic, learning, discrete time PI controller, rule base, neural network.

1- General Presentation.

Fuzzy logic, introduced by Zadeh, [Zadeh 73], has been applied with a certain success to process control since Mandani's pioneering work, [Mandani 74]. Today, many industrial applications use fuzzy control, especially in Japan, [Yasunobu 85], [Hakata 90].

A fuzzy controller looks like a PI or PID controller (P : Proportional, I : Integral, D : Derived). These types of command implicitely link the manipulated variables of a plant to its observed outputs. But while the classical automatician methods elaborate their commands from an explicit numerical model of the system to be controlled, a fuzzy controller reproduces the evaluation of a human operator.

Therefore, in many industrial processes, a skilled operator is able to manipulate correctly the different command organs, from the following informations :

- the difference between the plant outputs and the set point values,
- the change in error,
- the system delay,
- the foreseeable evolution of disruptive variables (for instance, in building energy management, solar gains or external temperature).

This operator's judgment can be characterized by the use of "rule of thumb" and multi-criterial choices. The basis of this judgment is often a miscellany of structured knowledge and intuition. Therefore, it is sometimes difficult to automate the extraction of the operator's knowledge. Here, the rule base is derived in a way based on learning.

2- Quick Presentation of Fuzzy Logic.

The Universe of discourse, **U**, is the space on which a variable is defined. A fuzzy subset of **U**, A, is characterized by a membership function $\mu_A : \mathbf{U} \rightarrow [0,1]$, which associates a number $\mu_A(\mathbf{u})$ in [0,1] to each element $\mathbf{u} \in \mathbf{U}$. This number $\mu_A(u)$ represents the membership degree of **u** to A. The notion of membership function is a generalization of the characteristic function of the A set, $1_A : \mathbf{U} \rightarrow \{0,1\}$. The fact of being able to classify a variable **u** in a continuous way from 0 to 1 (from totally false to totally true), allows to prevent sudden thresholds. The variation domain of **u** is divided into a number of fuzzy subsets generally noted PB (Positive Big), PS (Positive Small), ZR ZeRo), NM (Negative Medium) etc, the number of these subsets can vary depending on the required accuracy.

In the case of the following figure, The value $\mathbf{u_0}$ will be :

$$\mu_{PS}(\mathbf{u_0}) = 0.3 \quad \text{and} \quad \mu_{PM}(\mathbf{u_0}) = 0.7$$

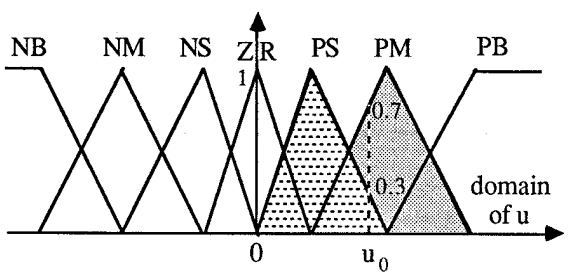


fig 1- triangular membership functions

A fuzzy logic controller is made of a set of rules like :

if "X is A" **then** "U is B"

where A and B are fuzzy subsets,
X is the observed system output,
U is the manipulated variable.

The fuzzy inference mechanism, which can be considered as a Generalised Modus Ponens, allows us to give a meaning to the consequent of a rule, even if the premise, "X is A", is not *exactly* true.

For a SISO system (Single Input Single Output), the input variables of the controller are often the difference, **e**, between the output system and a given set-point, and the change in error, **de**. More precisely, if e(k) is the error at step k, we have :

$$\mathbf{de}(k) = \mathbf{e}(k) - \mathbf{e}(k-1)$$

Calling **du**(k) the change in control, we build a PI-like controller, for instance, in writing :

$$\mathbf{u}(k) = \mathbf{u}(k-1) + \mathbf{du}(k)$$

with **du**(k) function of e(k) and e(k-1).

Given the observation (e_0, de_0) and a set **I** of rules, **I** = {1,2,...N} such as :

rule i : **if** "**e** is A_i" **and** "**de** is B_i " **then** "**du** is C_i"

we must evaluate the truth value of the premises. For rule i, this truth value, α_i, can be expressed in two ways :

$$\alpha_i = \mathrm{Min}(\mu_{Ai}(e_0), \mu_{Bi}(de_0))$$

or

$$\alpha_i = \mu_{Ai}(e_0) \cdot \mu_{Bi}(de_0) \qquad \text{for } i \in \mathbf{I}$$

We have chosen the first way, but the results would be the same in the second.

If, to simplify, the values C_i , for $i \in \mathbf{I}$, of the consequent of the rules are crisp values and not fuzzy subsets, a classical defuzzyfication method is the weighted average, or center of gravity method :

$$\mathbf{du}(e_0, de_0) = \frac{\sum \alpha_i C_i}{\sum \alpha_i}$$

The control action is made of four steps :

1- reading, at each time step, of **e** and **de**,
2- evaluation of the membership functions for each fuzzy subset defined on the domains of **e** and **de**,
3- evaluation of the premise of the rules. For two input variables, evaluated with regard to n fuzzy subsets, there are n^2 possible rules.
4- evaluation of the consequent (defuzzyfication).

By analogy with neural nets, we can adopt the scheme of figure 2 and we shall talk of "neurons", net inputs and outputs. The weights w_i represent the different C_i of the consequent of the rules.

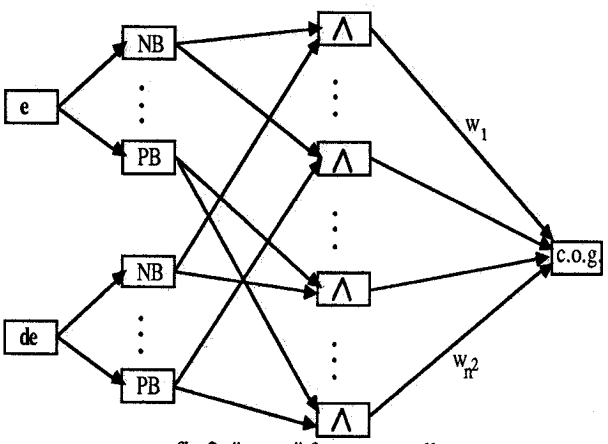


fig 2- "neuro" fuzzy controller.

The outputs of the units with a symbol "∧" are the minimum of their inputs and the output of the unit with the symbol "c.o.g." is the weighted average of its inputs. We shall call them respectively the **Min** neurons and the **c.o.g.** neuron.

3- Learning.

Knowledge extraction is often a major difficulty in a rule-based system. We shall develop a supervised learning method, allowing us to build rule basis in an automatic way.

We choose the case of a two inputs (**e** and **de**) and one output (**du**) controller. As we have seen, the set **I** of possible rules depends on the decomposition of **e** and **de** in fuzzy subsets. We suppose that there is a learning set, composed of m patterns : (e_i, de_i, du_i), $i \in \{1,...m\}$, with (e_i, de_i) : input stimulus, and du_i : desired response.

The weights w_i are initialised either to be zero, if we

start learning without any knowledge about the rules, or with approximative values.

When a pattern ($\mathbf{e}_i$, $\mathbf{de}_i$) is presented, the output net value, y_i , is :

$$y_i = \frac{\sum \alpha_k w_k}{\sum \alpha_k} \quad \text{for } k \in I$$

The error, y_i - $\mathbf{du}_i$, is a function of $(w_k)_{k \in I}$. The learning consists in finding a vector $W = (w_1,\ldots,w_N)^t$ which reduces this error on the learning set. One way to express the global error is :

$$E(W) = 1/2 \sum | y_i - \mathbf{du}_i |$$

We minimize E(W) by a gradient descent :

$$\Delta w_i = - \eta \frac{\partial E}{\partial w_i}$$

where η is a gain factor.

A simplification consists in computing $\partial E / \partial w_i$ at each presentation of ($\mathbf{e}_i$, $\mathbf{de}_i$, $\mathbf{du}_i$), and not after the processing of the whole learning set. It no longer means formally a gradient descent but rather a stochastic gradient descent, [Le Cun 85].

In this last case, if we call $\mathbf{n}_i$ the activity of the neuron i, (that is to say, the truth value of the premise implemented by neuron i), we have :

$$\frac{\partial E}{\partial w_i} = (y_i - \mathbf{du}_i) \frac{\mathbf{n}_i}{\sum \mathbf{n}_i}$$

the weights are therefore modified proportionally to the committed error and to the relative activity of the different neurons. The weight modification algorithm becomes :

$$\Delta w_i = - \eta (y_i - \mathbf{du}_i) \frac{\mathbf{n}_i}{\sum \mathbf{n}_i}$$

4- Rule Extraction.

Begining with a null or approximative knowledge, learning consists in determining the weights w_i which minimize E(W).

With the choice of triangular membership functions intersecting at 1/2 on the Y-axis, cf. fig.1, for a given observation ($\mathbf{e}_0$,$\mathbf{de}_0$), only four membership functions are different from zero, taking values :

α and 1-α for $\mathbf{e}_0$, with $\alpha \in [0,1]$

β and 1-β for $\mathbf{de}_0$, with $\beta \in [0,1]$

Therefore, four neurons **Min** , at the most, among N are non-zero, with the value $\alpha \wedge \beta$, $\alpha \wedge (1-\beta)$, $\beta \wedge (1-\alpha)$ and $(1-\alpha) \wedge (1-\beta)$. Besides, the biggest value of such a neuron is 1 and in this case the outputs of all the other neurons are zero.

If one of the neuron **Min** takes the value 1, the premise it represents is therefore verified and the value of the consequent is given by the weight which links this neuron with the output neuron. For instance, let's suppose that the k^{th} **Min** neuron gives the value 1 and corresponds to the premise :

"**e** is PB and **de** is NM"

We have therefore the rule :

if " **e** is PB" and " **de** is NM" then " **du** is w_k"

w_k being the weight linking the **Min** neuron to **c.o.g.** neuron.

5. Application to Building Regulation

We applied this method to the simulation of a building model.

5.1. The Used Model

This model is presented in the form of a two-order system :

$$dX(t) = F X(t) + G U(t)$$
$$T_{room} = H X(t) + K U(t)$$

where $X = (X_1,X_2)^t$ is the state vector,

$U = (P, E, T)^t$ is the control and disturbance vector, with P : heating power (Watt), E : solar radiation (W/m^2) and T : outside temperature (deg. C),

F, G, H and K are matrices.

The two time constants of this three inputs and one output system are :

$\tau_1 = 44.15$ hours

$\tau_2 = 1.55$ hour.

This model is derived from the reduction of a detailed model [Neirac 90]. The corresponding building is very glazed, a fact which will provoke overheating problems in case of important solar radiations.

5.2. The Learning Set and the Generalization Set

To obtain a training set, we have made use of a discrete time PI controller, approximatively tuned, operating with a fixed time step (1/4 h). Then the model with its controller has been simulated with real meteorological data. This has given a file containing a set of training samples $(\mathbf{e}_i, \mathbf{de}_i, \mathbf{du}_i)$, i=1 à 1920, corresponding to a period of 20 days, with :

$\mathbf{e}_i = T_{room} - T_{set\text{-}point}$ at time i (°C)
$\mathbf{de}_i = \mathbf{e}_i - \mathbf{e}_{i-1}$
$\mathbf{du}_i$ = change in control (Watt)
= desired response for input pattern $(\mathbf{e}_i, \mathbf{de}_i)$

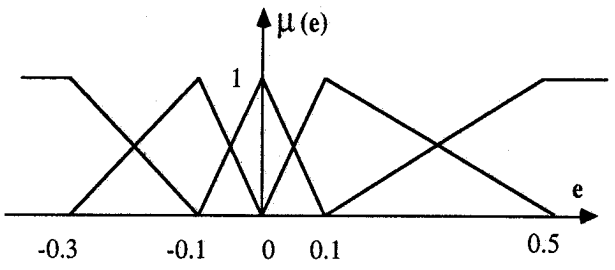
In our case, we had $\mathbf{e} \in [-0.52, 5.6]$, $\mathbf{de} \in [-0.64, 0.60]$ and $\mathbf{du} \in [-256,190]$. The large variations of **e** (overheatings) were caused by important solar gains, not for lack of control.

This file can be divided into two parts : the first for learning (learning set) and the second for testing the generalization capability of the fuzzy controller (generalization set). When using the generalization set, the weights must be locked.

5.3. The Simulations

The learning set was constituted by the 480 first patterns and the generalization set by the 1440 others.

The variation domains of **e** and **de** were split into five fuzzy subsets noted NB, NS, ZR, PS, PB.



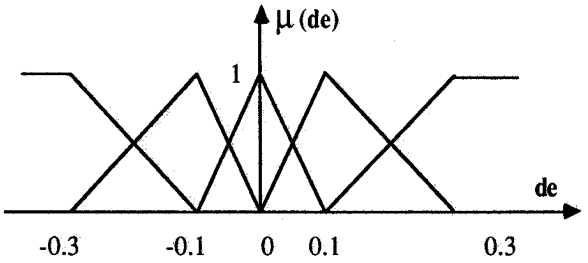


fig 3- fuzzy subsets for **e** and **de**.

With five fuzzy subsets, there are $5^2 = 25$ possible rules and hence 25 weights to determine. The learning was realized without initial knowledge (all the weights were initialized at zero), with $\eta = 0.1$. Each pattern was randomly presented to the fuzzy network of fig. 2. The error between the actual output, y_i, and the desired response, $\mathbf{du}_i$, was computed and the weights updated at each presentation.

After presentations of the 480 patterns of the learning set, we computed the learning error LE :

$$LE = \frac{1}{480} \sum_{i=1}^{480} |y_i - \mathbf{du}_i|$$

and began again as long as LE was to large. The fig. 4 shows the evolution of LE as a function of the number of presentations of the whole learning set.

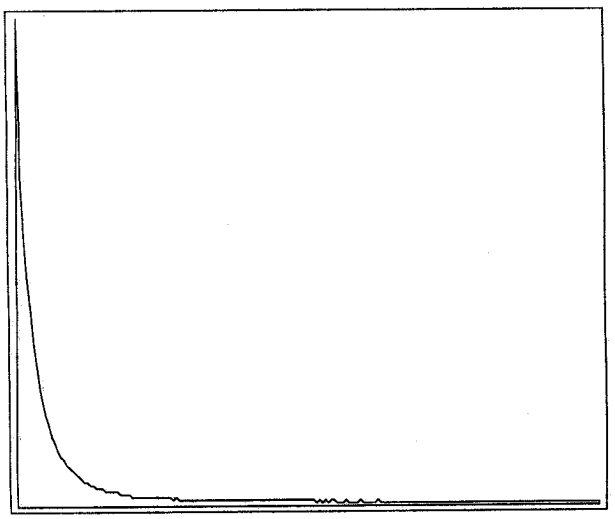


fig 4- evolution of learning error.

We stopped the learning when LE was below 2.6 Watts, after 100 presentations. Then the weights were locked and the generalization patterns were presented once, with as result :

$$\frac{1}{1440}\sum_{i=1}^{1440}|y_i - du_i| = 2.85 \text{ Watts}$$

The generalization capability of the net is quite good.

Identified weights are given in table 1.

e \ de	NB	NS	ZR	PS	PB
NB	165	125	100	85	20
NS	30	40	23	1	-19
ZR	43	25	0	-32	-30
PS	46	-1	-24	-30	-31
PB	-107	-149	-156	-211	-239

table 1 - weights after learning.

Remark : the grey squares represent not very frequent cases. Their values, hence, are not significant. The table can be read as follows :

if "**e** is NB" **and** "**de** is PS" **then** "**du** = 85 Watts"

We can compare this identified table to an "a priori" table of rule for a PI-like controller :

e \ de	NB	NS	ZR	PS	PB
NB	PB	PB	PB	PS	ZR
NS	PB	PS	PS	ZR	NS
ZR	PB	PS	ZR	NS	NB
PS	PS	ZR	NS	NS	NB
PB	ZR	NS	NB	NB	NB

table 2- "a priori" rule base for a PI-like controller.

Those two tables are in accordance, except the two cases already mentioned.

We have used this rule base, with identified weights, to assure the regulation of our model. In figure 5, the higher curve shows the variations of room temperature, and the lower, the variations of the heating power. The solar radiation occasionally creates some overheatings, by lack of ventilation system, but apart from these periods, the control is completely satisfactory and is as accurate as the initial PI control.

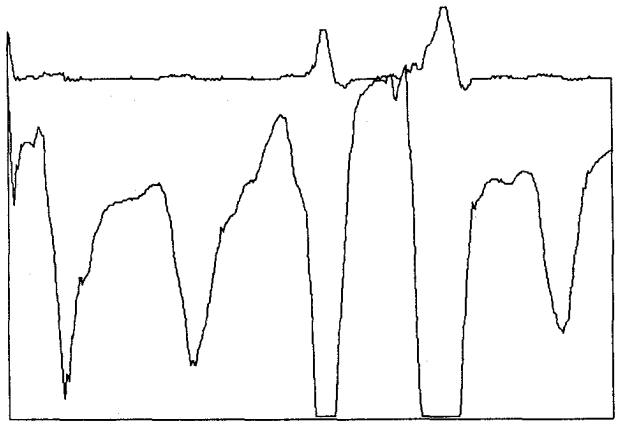


fig 5- simulated behaviour of the fuzzy controller, with identified rules.

6- Conclusion.

If we have a learning set, the proposed fuzzy controller is able to learn the necessary rules to assure building regulation. It must be noted that the learning period lasts 4 or 5 days, which is comparable to the time taken to do identification by the least recursive square method.

The following stage will consist in developing an in-line self-tuning algorithm, taking into account the foreseeable evolution of solar radiation which is the most important disruptive variable in our case.

7- References

[Hakata 90] Hakata, T. and Masuda, J. "Fuzzy control of cooling system utilizing heat storage". *Proc. of Int. Conf. on Fuzzy Logic and Neural Netwoks. Iizuka'90*, Japan,1990.

[Le Cun 85] Le Cun, Y. "Une procédure d'apprentissage pour réseaux à seuil asymétriques". *Proc. of Cognitiva*, p. 599-604, 1985.

[Mamdani 74] Mamdani, E.H. "Application of fuzzy algorithms for control of a simple dynamic plant". *Proceedings of IEE 121-12*, p. 1585-1588, 1974.

[Neirac 90] Neirac, F. Personnal communication.

[Yasanobu 85] Yasanobu, S. and Miyamoto S. "Automatic train operation system by predictive fuzzy control". In "*Industrial application of fuzzy control*", Sugeno Ed., North Holland, 1985.

[Zadeh 73] Zadeh, L.A. "Outline of a new approach to the analysis of complex systems and decision processes". *IEEE trans. on SMC*, p 28-44, 1973.